“华为杯”第十五届中国研究生

数学建模竞赛

题 目　　机场新增卫星厅对中转旅客影响的评估方法

摘　　要：

针对题目要求评估机场新增卫星厅对中转旅客的影响，本文对三种情形的航班－登机口分配问题建立**组合优化**数学模型，并通过求解这些模型进行数据评估分析。

在题目给出的数据文件中，对 20 日到达或 20 日出发的航班记录、旅客中转记录进行提取和统计，结果如下：有效航班记录共 **303** 对，其中宽体机数量 49 对，窄体机数量 **254** 对；有效旅客中转记录共 1703 条，包涵总人数 **2833** 人；登机口共 69 个，其中航站楼 T 有 **28** 个登机口，卫星厅 S 有 **41** 个登机口。

针对问题 1，要求尽可能多地分配航班并在此基础上最小化使用登记口的数量。通过使用**遗传算法**求解，并引入**模拟退火**的思想设计适应度函数，求出了航班－登机口的最优分配方案。分配方案统计如下：共 254 对航班分配到登机口，其中宽体机数量 49 对，成功率 100%，窄体机数量 205 对，成功率 80.71%，见图 3-2、图 3-3 所示；被使用登机口 66 个，其中航站楼 T 被使用 27 个，使用总时间 20595 分钟，平均使用率 52.97%，卫星厅 S 被使用 39 个，使用总时间 34860 分钟，平均使用率 62.07%，见图 3-4、图 3-5 所示（**本文所有计算平均使用率的过程中，最小空挡间隔时间 45 分钟不计入登机口被使用时间**）。

针对问题 2，要求在问题 1 的基础上最小化中转旅客的总体最短流程时间。通过修改适应度函数，在目标中添加中转旅客的总体最短流程时间这一因素，求出了最优分配方案。分配方案统计如下：共 254 对航班分配到登机口，其中宽体机数量 49 对，成功率 100%，窄体机数量 205 对，成功率 80.71%，见图 4-1、图 4-2 所示；被使用登机口 67 个，其中航站楼 T 被使用 28 个，使用总时间 22340 分钟，平均使用率 55.41%，卫星厅 S 被使用 39 个，使用总时间 33185 分钟，平均使用率 59.09%，见图 4-3、图 4-4 所示；换乘失败旅客数为 0，失败率 0%（不考虑分配在临时机位的航班）。

针对问题 3，要求在问题 2 的基础上最小化换乘旅客总体紧张度。通过引入旅客换乘连接变量，计算旅客换乘紧张度，将总体紧张度加入目标函数，并设计了新的适应度函数，求出了最优分配方案。分配方案统计如下：共 253 对航班分配到登机口，其中宽体机数量 49 对，成功率 100%，窄体机数量 204 对，成功率 80.31%，见图 5-1、图 5-2 所示；被使用登机口 66 个，其中航站楼 T 被使用 27 个，使用总时间 22165 分钟，平均使用率 57.01%，卫星厅 S 被使用 39 个，使用总时间 33385 分钟，平均使用率 59.45%，见图 5-3、图 5-4 所示；旅客 1882 人换乘成功，951 人的航班分配至临时机位，换乘失败旅客数为 0，失败率 0%（不考虑分配在临时机位的航班）；以 5 分钟**离散化**统计旅客的换乘时间，旅客换乘时间分布图如图 5-5 所示；以 0.1 离散化统计旅客的换乘紧张度，旅客换乘紧张分布图如图 5-6 所示。

关键字：组合优化　遗传算法　模拟退火　区域离散化

目　　录

1．问题重述

1.1　问题背景

由于旅行业的快速发展，某航空公司在某机场的现有航站楼 T 的旅客流量已达饱和状态，为了应对未来的发展，现正增设卫星厅 S。但引入卫星厅后，虽然可以缓解原有航站楼登机口不足的压力，对中转旅客的航班衔接显然具有一定的负面影响。

飞机在机场廊桥（登机口）的一次停靠通常由一对航班（到达航班和出发航班，也叫“转场”）来标识，如图 1-1 所示。航班－登机口分配[1]就是把这样的航班对分配到合适的登机口。所谓的中转旅客就是从到达航班换乘到由同一架或不同架飞机执行的出发航班的旅客。

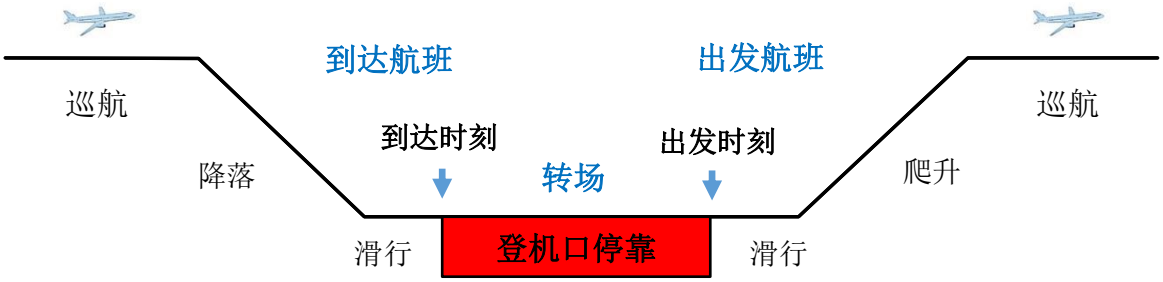

图 1-1　航班及其停靠示意图

单纯的航班－登机口的优化分配问题已经被很好地解决，但在优化分配登机口的同时考虑最小化旅客行走时间，学界研究有限，市场上产品一般也不具备此一功能。因此建立合适的数学模型，优化分配登机口，分析中转旅客的换乘紧张程度，为航空公司航班规划的调整提供参考依据就变得非常重要。

1.2　需要解决的问题

本赛题要求就以下三种情形对航班－登机口分配问题建立数学优化模型，并通过求解这些模型，进行数据评估分析。

问题 1：本题只考虑航班－登机口分配。作为分析新建卫星厅对航班影响问题的第一步，首先要建立数学优化模型，尽可能多地分配航班到合适的登机口，并且在此基础上最小化被使用登机口的数量。本问题不需要考虑中转旅客的换乘，要求把建立的数学模型进行编程，并求得最优解。

问题 2：考虑中转旅客最短流程时间。本问题是在问题 1 的基础上加入旅客换乘因素，要求最小化中转旅客的总体最短流程时间，并且在此基础上最小化被使用登机口的数量。本题不考虑旅客乘坐捷运和步行时间，要求编程并求得最优解。

问题 3：考虑中转旅客的换乘时间。如前所述，新建卫星厅对航班的最大影响是中转旅客换乘时间的可能延长。因此，数学模型最终需要考虑换乘旅客总体紧张度的最小化，并且在此基础上最小化被使用登机口的数量。本问题可以在问题 2 的基础上细化，引入旅客换乘连接变量，并把中转旅客的换乘紧张度作为目

标函数的首要因素。本问题要求把建立的数学模型进行编程，并求得最优解。

1.3 相关描述

（1）机场布局：航站楼 T 和卫星厅 S 的布局设计如 1-2 图所示。T 具有完整的国际机场航站功能，包括出发、到达、出入境和候机。卫星厅 S 是航站楼 T 的延伸，可以候机，没有出入境功能。T 和 S 之间有捷运线相通，可以快速往来运送国内、国际旅客。航站楼 T 有 28 个登机口，卫星厅 S 有 41 个登机口。

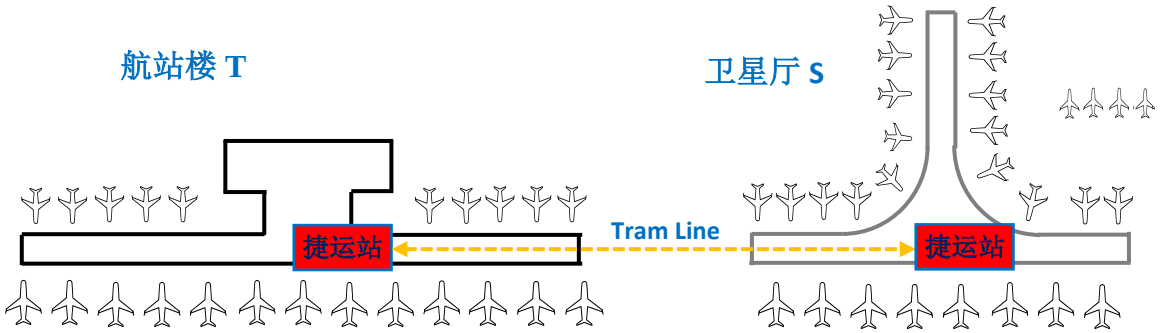

图 1-2　卫星厅 S 相对于航站楼 T 示意图

（2）登机口分配：登机口属于固定机位，配置有相应的设备，方便飞机停靠时的各种技术操作。航班一登机口的分配需要考虑如下规则：

I．T 和 S 的所有登机口统筹规划分配；

II．每个登机口的国内/国际、到达/出发、宽体机/窄体机等功能属性事先给定，不能改变，飞机转场计划里的航班只能分配到与之属性相吻合的登机口；

III．每架飞机转场的到达和出发两个航班必须分配在同一登机口进行，其间不能挪移别处；

IV．分配在同一登机口的两飞机之间的空挡间隔时间必须大于等于 45 分钟；

V．机场另有简易临时机位，供分配不到固定登机口的飞机停靠。

2．模型假设及符号说明

2.1 模型假设

（1）假设一：所有建模和数据分析都是针对航站楼 T 和卫星厅 S 同时使用的情形；

（2）假设二：假定旅客无需等待捷运，随时可以发车，单程一次需要 8 分钟；

（3）假设三：临时机位数量无限制。

2.2 符号说明

符号	符号含义
n	待分配航班数量
m	登机口数量
f_i	第 i 对航班，$i=1,2,\cdots,n$
g_j	第 j 个登机口，$j=1,2,\cdots,m$
a_i	航班 i 的到达时刻
d_i	航班 i 的出发时刻
x_{ij}	若航班 i 分配到登机口 j，则 $x_{ij}=1$，否则 $x_{ij}=0$
f_i^{Type}	航班 i 的机体类别，可为 N（窄体机）或 W（宽体机）
g_j^{Type}	登机口 j 的机体类别，可为 N（窄体机）或 W（宽体机）
f_i^{Nation}	航班 i 转场的＜到达类型，出发类型＞
g_j^{Nation}	登机口 j 转场的＜到达类型，出发类型＞
α	同登机口相邻航班最小空挡间隔时间，本文为 45 分钟
R_i	与航班 i 登机口占用时间存在冲突的飞机集合
y_j	若登机口 j 未分配到航班，则 $y_j=1$，否则 $y_j=0$
N	种群规模
$fitness$	个体的适应度函数
r	随机数
λ,β	问题 1 的数值参数
h	旅客数量
t_u	第 u 位旅客，$u=1,2,\cdots,h$
t_u^v	第 u 位旅客在中转情况为 v 下的最短流程时间
$tensity_u$	第 u 位旅客的换乘紧张度

3．问题 1 的模型建立与求解

3.1 问题 1 的分析

问题 1 不需要考虑中转旅客的换乘，仅考虑航班 $F=\{f_1,f_2,\cdots,f_n\}$ 与登机口 $G=\{g_1,g_2,\cdots,g_m\}$ 的分配。为了尽可能多地分配航班到合适的登机口，并且在此基础上最小化被使用登机口的数量，要在满足硬约束条件下，以最大化分配到登机口的航班数量和未使用登机口数量为目标函数，从而求解问题 1。需要满足的硬约束如下：

（1）唯一性约束：一对航班仅能够停靠一个登机口，这里可使用 0—1 二维矩阵来表示登机口占用情况；

（2）机体类别匹配约束：航班与其停靠登机口的机体类别一致；

（3）转场航班的到达类型、出发类型匹配约束：航班的到达、出发类型要与登机口可转场的航班类型匹配；

（4）独占性约束：一个登机口不能同时停靠两对航班；

（5）同一登机口安全时间间隔约束：对于分配到同一登机口的两对航班空挡间隔时间必须大于等于 45 分钟。

3.2 模型的准备与建立

3.2.1 数据定义

（1）常量

α ——同一登机口相邻航班的最小空挡间隔时间，本文取值为 45 分钟。

（2）变量

f_i ——第 i 对航班，$i=1,2,\cdots,n$

g_j ——第 j 个登机口，$j=1,2,\cdots,m$

a_i ——航班 i 的到达时刻；

d_i ——航班 i 的出发时刻；

x_{ij} ——若航班 i 分配到登机口 j，则 $x_{ij}=1$，否则 $x_{ij}=0$；

f_i^{Type} ——航班 i 的机体类别，其可能取值为 N（窄体机），W（宽体机）；

g_j^{Type} ——登机口 j 的机体类别，其可能取值为 N（窄体机），W（宽体机）；

f_i^{Nation} ——航班 i 转场的＜到达类型，出发类型＞，其可能取值为 $<D,D>$，$<D,I>$，$<I,D>$，$<I,I>$；

g_j^{Nation} ——登机口 j 转场的＜到达类型，出发类型＞，其可能取值为 $<D,D>$，$<I,I>$，$<D,\{D,I\}>$，$<I,\{D,I\}>$，$<\{D,I\},I>$，$<\{D,I\},D>$，$<\{D,I\},\{D,I\}>$；

（3）中间变量

R_i ——与航班 i 登机口占用时间存在冲突的飞机集合：$\forall j\in\{1,2,\cdots,n\}$，若 $a_i<a_j\cap d_i+\alpha>a_j$ 或 $a_i>a_j\cap a_i<d_j+\alpha$，则 $a_j\in R_i$，即检测与航班 i 时间存在冲突或间隔小于最小空挡间隔时间 α 的飞机放入 R_i。

对一对航班的<到达类型，出发类型>与登机口的<到达类型，出发类型>进行类型编号，在求解过程中只有登机口 j 的可停靠类型包含航班 i 的停靠类型，x_{ij} 才有可能取 1，不同类型登机口可停靠的航班转场类型如表 3-1 所示。

表 3-1　登机口可停靠的飞机转场类型

登机口转场类型	可停靠的航班转场类型
$<D,D>$	$<D,D>$
$<I,I>$	$<I,I>$
$<D,\{D,I\}>$	$<D,D>$，$<D,I>$
$<I,\{D,I\}>$	$<I,D>$，$<I,I>$
$<\{D,I\},I>$	$<D,I>$，$<I,I>$
$<\{D,I\},D>$	$<D,D>$，$<I,D>$
$<\{D,I\},\{D,I\}>$	$<D,D>$，$<D,I>$，$<I,D>$，$<I,I>$

3.2.2　问题 1 的模型建立

（1）目标函数：由问题 1 可知，为了尽可能多地分配航班到合适的登机口，并且在此基础上最小化被使用登机口的数量，因此本文以最大化分配到登机口的航班数量和未使用登机口 y_j 的数量为目标进行建模，目标函数如下：

$$\max\ z=\sum_{i=1}^{n}\sum_{j=1}^{m}x_{ij}+\sum_{j=1}^{m}y_j \tag{3-1}$$

（2）唯一性约束，即一对航班仅能够停靠一个登机口：

$$\sum_{j=1}^{m}x_{ij}\le 1\ ,\ i=1,2,\cdots,n \tag{3-2}$$

（3）机体类别匹配约束，即航班 i 与其停靠登机口 j 的机体类别一致：

$$f_i^{Type}=g_j^{Type}\ ,\ i=1,2,\cdots,n\ ,\ j=1,2,\cdots,m \tag{3-3}$$

（4）转场航班 i 与停靠登机口 j 的到达类型、出发类型匹配约束，即登机口 j 的可停靠类型包含航班 i 的停靠类型：

$$f_i^{Nation}\in g_j^{Nation}\ ,\ i=1,2,\cdots,n\ ,\ j=1,2,\cdots,m \tag{3-4}$$

（5）独占性约束，一个登机口 j 不能同时停靠两对航班 i,k；同一登机口安全时间间隔约束，对于分配到同一登机口 j 的两对航班 i,k 的空挡间隔时间必须大于等于 45 分钟。即在冲突集合 R_i 中的任意两对航班 i,k 不能停靠在一个登机口：

$$\sum_{k\in R_i}x_{ij}+x_{kj}\le 1\ ,\ i,k=1,2,\cdots,n\ ,\ j=1,2,\cdots,m \tag{3-5}$$

（5）综上，可以建立问题 1 的数学模型，见 $MOD1$。

*MOD*1

$$\max\ z_1 = \sum_{i=1}^{n}\sum_{j=1}^{m} x_{ij} + \sum_{j=1}^{m} y_j$$

$$s.t.\begin{cases} \sum_{j=1}^{m} x_{ij} \le 1\ ,\ i = 1,2,\cdots,n \\ f_i^{Type} = g_j^{Type}\ ,\ i = 1,2,\cdots,n\ ,\ j = 1,2,\cdots,m \\ f_i^{Nation} \in g_j^{Nation}\ ,\ i = 1,2,\cdots,n\ ,\ j = 1,2,\cdots,m \\ \sum_{k \in R_i} x_{ij} + x_{kj} \le 1\ ,\ i,k = 1,2,\cdots,n\ ,\ j = 1,2,\cdots,m \end{cases}$$

3.3 模型的求解与分析

问题 1 模型的特点及算法引入：登机位分配问题是一个 NP 难的组合优化问题（在多项式的计算时间内无法获得最优解），当扩大求解规模后，可行解的数量将呈指数级增长，若仍然采用精确算法求解该问题时会出现维数灾难[2]。美国 Michigan 大学的 Holland 教授提出的遗传算法（Genetic Algorithm，GA）是求解复杂组合优化问题的有效方法，是求解全局最优化问题的一种新型算法，它是模仿生物进化与遗传原理而设计的一类随机搜索的优化方法，已证明带精英保留的遗传算法依概率 1 收敛到全局最优解[3]。

3.3.1 算法设计

遗传算法的基本思想是从初始种群出发，通过选择、交叉和变异等操作，迭代产生新的子代，并以个体适应度为指导，不断逼近解区间内的最优解[4]。我们采用了遗传算法来求解问题 1，并引入模拟退火思想设计适应度函数，模拟退火思想能够加快收敛速度并最终趋于稳定。算法分为如下的步骤进行求解：

STEP1　个体编码设计

采用实数编码，用长度为 n 的个体记录 n 对航班分配到的登机位。如 3 2 1 4 2 3 4 4 表示 8 对航班在 4 个登机位上的分配方案：第 1 对航班分配至 3 号登机位，第 2 对航班分配至 2 号登机位……。

STEP2　初始种群生成

在登机口分配过程中，配对航班要满足模型中所有约束条件，如果随机产生个体，该个体成为可行解的可能性很小，故在初始种群的生成过程中通过约束来控制，以保证其成为可行解，个体生成步骤如下：

I．随机产生配对航班的优先级顺序，即产生 $1\sim n$ 的整数数组 p 并对其随机排序，则数组 p 为航班分配优先级顺序；

II．按优先级为每对航班分配登机口，记 $i=1$；

III．对航班 p_i，依次检测是否与某一登机口满足所有约束条件，若第 j 个登机口可以匹配航班 p_i，则将个体第 p_i 位取值为 j，若没有登机口可与 p_i 匹配，则将个体第 p_i 位取值为 0；

IV．若 $i<n$ 成立，$i=i+1$，转 III，否则输出个体编码。

此算法需要对每一个登机位的空闲时间实时更新，设定初始种群规模为 $N=1000$，即循环此步 1000 次以产生初始种群。

STEP3　赌轮选择

赌轮选择的基本思想是生存力越好的个体被选择的概率越大，是一种有放回的随机采样方法。赌轮选择根据个体的适应度 $fitness$ 计算个体在种群中出现的概率，按照出现概率随机选择个体构成子代。步骤如下：

I．计算每个个体的适应度 $fitness(i)$，$i=1,2,\cdots,N$；

II．计算个体被遗传到下一代的概率：

$$P_i=\frac{fitness(i)}{\sum_{j=1}^{N} fitness(j)} \tag{3-6}$$

III．计算个体的累积概率：

$$P_i=\frac{\sum_{k=1}^{i} fitness(k)}{\sum_{j=1}^{N} fitness(j)} \tag{3-7}$$

IV．随机产生$[0,1]$区间内的一个随机数 r，若 $P_{i-1}\le r<P_i$，则个体 i 被选择。

在每代种群出现的累积概率，通过产生随机数的方法选择个体进入下一代群体中。这样适应度函数的设计就显得十分重要，直接采用目标函数值 z 作为适应度函数，不同分配结果取得的值差别不大，可能导致算法的收敛速度过慢，这里我们引入模拟退火思想[5]，重新设计适应度函数为：$fitness=e^{\lambda z_1}\times 10^{\beta}$，其中 z_1 为目标函数值，$\lambda=0.02$，$\beta=3$为参数值。

STEP4　两点交叉

本文采用两点交叉算子，首先由交叉概率 P_c 选择进行交叉的两个个体作为父代，然后随机产生两个交叉点，最后对父代交换两个交叉点之间的登机位号产生子代，交叉示意图如图 3-1 所示。

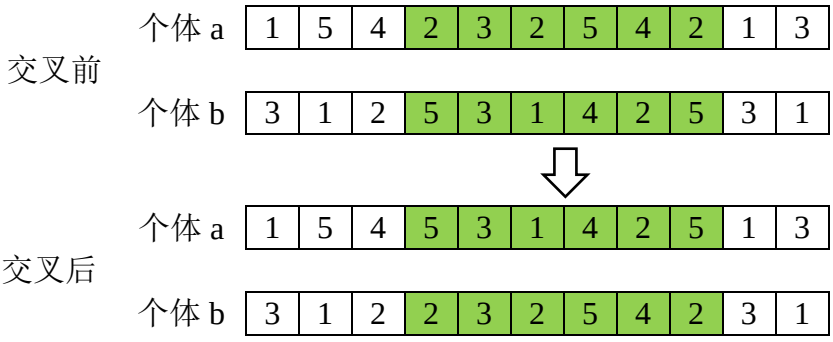

图 3-1　交叉操作示意图

STEP5　变异算子

由于产生子代进行的交叉操作均为随机产生操作点，子代极有可能不满足上文所述的约束条件，故通过变异操作使其成为可行解。变异的具体步骤如下：对于交叉产生的子代个体，顺序检查交叉起点之后的所有编码，若满足约束条件则该位编码不变，否则按初始种群生成的规则重新为当前编码位取值。

STEP6　精英保留

在产生的子代种群中，选出适应度最高的个体与当前最优个体（精英）进行比较，适应度较高的成为新的最优个体（精英）。

STEP7　终止条件

若满足终止条件（种群进化 100 代），则算法停止，否则返回 STEP3。

3.3.2　问题 1 求解结果及分析

对于问题 1 的求解，首先对数据进行预处理：（1）保留 20 日到达或 20 日出发的航班记录，共 303 条；（2）到达时刻与出发时刻转换为距 0 点的相对时间，单位为分钟；（3）对航班与登机口的＜到达类型，出发类型＞进行数值编号，编号表如表 3-2 所示。

表 3-2　＜到达类型，出发类型＞的数值编号

＜到达类型，出发类型＞	数值编号	＜到达类型，出发类型＞	数值编号
$<D,D>$	0	$<D,\{D,I\}>$	5
$<D,I>$	1	$<\{D,I\},D>$	6
$<I,D>$	2	$<\{D,I\},I>$	7
$<I,I>$	3	$<\{D,I\},\{D,I\}>$	8
$<I,\{D,I\}>$	4		

按照 3.3.1 节所述的算法流程，我们进行了遗传算法的编程并求出了最优结果。由数据预处理及统计可知，航班记录共 303 对，其中 49 对为宽体机，254 对为窄体机；登机口共 69 个，其中航站楼 T 有 28 个登机口，卫星厅 S 有 41 个登机口。计算结果表明：

（1）在 303 对航班记录中，有 254 对航班成功分配登机口，其中宽体机有 49 对航班分配成功，窄体机有 205 对航班分配成功，宽体机航班的分配成功率为 100%，窄体机航班的分配成功率为 80.71%，总成功率 83.83%。

全部航班记录、窄体机航班、宽体机航班的成功分配数量如图 3-2 所示，其中每一柱由上至下的 2 个数字代表该类型总量、成功分配数量；全部航班记录、窄体机航班、宽体机航班的成功分配率如图 3-3 所示，其中每一柱由上至下的 2 个数字代表该类型总量、分配成功率。

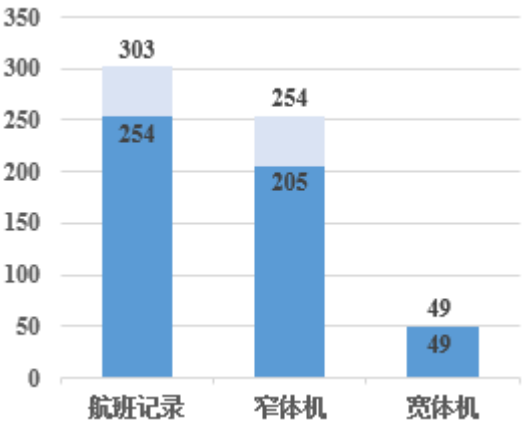

图 3-2　全部航班记录、窄体机航班、宽体机航班的成功数量

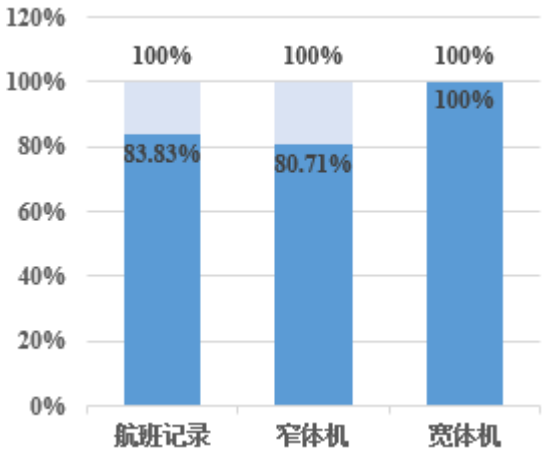

图 3-3　全部航班记录、窄体机航班、宽体机航班的成功率

（2）254 对成功分配的航班共停靠在了 66 个登机口，其中航站 T 楼使用了 27 个登机口，卫星厅 S 使用 39 个登机口；

（3）由计算结果统计可得航站楼 T 使用的 27 个登机口共被使用 20595 分钟，平均使用率为 52.97%，卫星厅 S 使用的 39 个登机口共被使用 34860 分钟，平均使用率为 62.07%，故总使用的 66 登机口共被使用 55455 分钟，平均使用率为 58.35%。平均占有虑的计算公式如下（**本文所有计算平均使用率的过程中，最小空挡间隔时间 45 分钟不计入登机口被使用时间**）：

$$\text{平均使用率} = \frac{\text{登机口被使用的时间之和}}{\text{被使用登机口数量} \times \text{一天总时长}} \tag{3-8}$$

问题 1 数据预处理采用 Java 编写程序，程序文件和预处理结果详见论文附件。问题 1 求解采用 C++编写主程序，程序文件详见附附件，程序模块介绍见附录 1。

总登机口、T 登机口和 S 登机口的使用数量如图 3-4 所示，其中每一柱由上至下的 2 个数字代表该类型总量、使用数量；总登机口、T 登机口和 S 登机口的平均使用率如图 3-5 所示，其中每一柱由上至下的 2 个数字代表该类型总量、平均使用率。

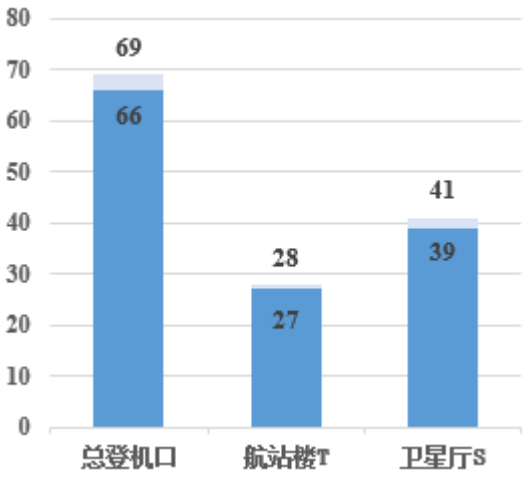

图 3-4　总登机口、T 登机口和 S 登机口的使用数量

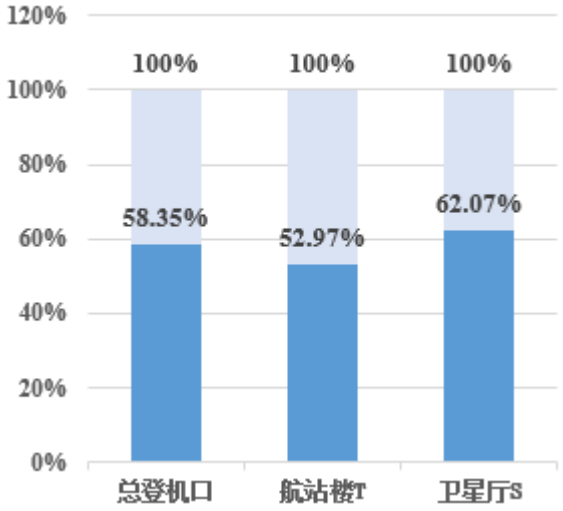

图 3-5　总登机口、T 登机口和 S 登机口的平均使用率

4. 问题 2 的模型建立与求解

4.1 问题 2 的分析

问题 2 是在问题 1 的基础上要求考虑中转旅客$T=\{t_1,t_2,\cdots,t_h\}$的最短流程时间，目标是最小化中转旅客的总体最短流程时间，并且在此基础上最小化被使用登机口的数量。问题 2 不需要考虑旅客乘坐捷运和步行时间。因此问题 2 的航班分配首先也需要满足所有硬约束条件：唯一性约束；机体类别匹配约束；转场航班的到达类型、出发类型匹配约束；独占性约束；同一登机口安全时间间隔约束。

此外，针对旅客的中转记录，首先要匹配其<到达类型，出发类型>，然后根据该旅客的到达航班、出发航班所分配的航站楼查表找到其对应的最短流程时间，以最大化分配到登机口的航班数量、最小化中转旅客的总体最短流程时间和最大化未使用登机口数量为目标函数，从而求解问题 2。

中转旅客从前一航班的到达至后一航班的出发之间的流程称为旅客流程，按国内（D）和国际（I）、航站楼（T）和卫星厅（S）组合成 16 种不同的场景。这些场景的最短流程时间如表 4-1 所示。

表 4-1 不同场景的最短流程时间

出发 / 到达		国内出发（D）		国际出发（I）	
		航站楼 T	卫星厅 S	航站楼 T	卫星厅 S
国内到达（D）	航站楼 T	15	20	35	40
	卫星厅 S	20	15	40	35
国际到达（I）	航站楼 T	35	40	20	30
	卫星厅 S	40	45	30	20

4.2 模型的准备与建立

4.2.1 数据定义

（1）变量

t_u——第u位旅客，$u=1,2,\cdots,h$

t_u^v——第u位旅客在中转情况为v的最短流程时间，$u=1,2,\cdots,h$

（2）t_u^v的取值定义及示例

下标u代表第u位旅客，$u=1,2,\cdots,h$。上标v代表旅客的中转场景类型，v由两位数字组成：第一位数字记录旅客的<到达类型，出发类型>，其定义同问题 1 中表 3-2，旅客中转的<到达类型，出发类型>可能情况为$<D,D>$，$<D,I>$，$<I,D>$，$<I,I>$，查表可知分别对应数字 0,1,2,3；第二位数字记录旅客到达、出发航班所在的航站楼类型，对应关系定义如表 4-2 所示。

表 4-2　旅客到达、出发航班停靠登机位的数值编号

数值编号	到达航班所在航站楼	出发航班所在航站楼
0	T	T
1	T	S
2	S	T
3	S	S

示例如下：假设下表是第 1 位旅客的中转信息，其中到达航班被分配在航站楼 T，出发航班被分配在航站楼 S，查表可知到达航班的到达类型为 I，出发类型为 D，由此可得$t_1^{21}=40$。

表 4-3　第 1 位旅客的中转信息

旅客记录号	乘客数	到达航班	到达日期	出发航班	出发日期
T0001	1	NV677	19-Jan-18	NV3446	19-Jan-18

4.2.2　问题 2 的模型建立

问题 2 需满足的约束条件同问题 1：

（1）唯一性约束：$\sum_{j=1}^{m} x_{ij} \le 1$，$i=1,2,\cdots,n$

（2）机体类别匹配约束：$f_i^{Type}=g_j^{Type}$，$i=1,2,\cdots,n$，$j=1,2,\cdots,m$

（3）转场航班i与停靠登机口j的到达类型、出发类型匹配约束：$f_i^{Nation} \in g_j^{Nation}$，$i=1,2,\cdots,n$，$j=1,2,\cdots,m$

（4）独占性约束：$\sum_{k\in R_i} x_{ij}+x_{kj} \le 1$，$i,k=1,2,\cdots,n$，$j=1,2,\cdots,m$

问题 2 的目标要最大化分配到登机口的航班数量、最小化中转旅客的总体最短流程时间和最大化未使用登机口数量，故目标函数要考虑以下因素：

（1）目标函数的统一：对于最小化中转旅客的总体流程时间$\sum_{u=1}^{h} t_u^v$，只需取反为$-\sum_{u=1}^{h} t_u^v$，即转变为最大化，做到与其它两个目标的统一；

（2）数量级的分析：通过对数据的观察，旅客的数量级为数千级，故中转的总体流程时间数量级为数万级，而分配到的航班数量为数百级，未使用登机口数量为数十级，所以为了平衡各个目标的数量级和优先级，设计目标函数如下：

$$z_2=\omega\sum_{i=1}^{n}\sum_{j=1}^{m} x_{ij}-\xi\sum_{u=1}^{h} t_u^v+\rho\sum_{j=1}^{m} y_j$$

其中ω，ξ，ρ为数量级调节参数（本文取$\omega=100$，$\xi=0.001$，$\rho=0.1$）。

综上，可以建立问题 2 的数学模型，见$MOD2$。

$$MOD2$$

$$\max\ z_2 = \omega\sum_{i=1}^{n}\sum_{j=1}^{m}x_{ij} - \xi\sum_{u=1}^{h}t_u^v + \rho\sum_{j=1}^{m}y_j$$

$$s.t.\begin{cases}\sum_{j=1}^{m}x_{ij} \le 1\ ,\ i=1,2,\cdots,n\\ f_i^{Type} = g_j^{Type}\ ,\ i=1,2,\cdots,n\ ,\ j=1,2,\cdots,m\\ f_i^{Nation} \in g_j^{Nation}\ ,\ i=1,2,\cdots,n\ ,\ j=1,2,\cdots,m\\ \sum_{k\in R_i}x_{ij}+x_{kj} \le 1\ ,\ i,k=1,2,\cdots,n\ ,\ j=1,2,\cdots,m\end{cases}$$

4.3 模型的求解及分析

问题 2 的数学模型与问题 1 类似，问题 2 仅在目标函数增加了中转旅客最短流程时间，增大了计算量。问题 1 采用的遗传算法仍然使用于问题 2，故我们仍采用遗传算法。

4.3.1 算法设计

问题 2 同样采用遗传算法求解，算法的流程与问题 1 相似，分为个体编码设计、初始种群生成、赌轮选择、两点交叉、变异算子，其中问题 1 赌轮选择中的适应度函数不再适用问题 2，设计新的适应度函数以考虑中转旅客的最短流程时间，适应度函数为 $fitness = z_2$，其中 z_2 为问题 2 的目标函数值。

4.3.2 问题 2 求解结果及分析

对于问题 2 的求解，首先对数据进行预处理：（1）首先保留 20 日到达或 20 日出发的旅客中转记录，然后删除航班不存在的无效记录，得到有效记录共 1703 条；（2）查找到达航班、出发航班的航班编号并记录；（3）查找到达航班、出发航班的＜到达类型，出发类型＞，并按表 3-2 记录其对应的数值编号。

问题 2 数据预处理采用 Java 编写程序，程序文件和预处理结果详见论文附件。

按照 4.3.1 节所述的算法流程，我们进行了遗传算法的编程并求出了最优结果。由数据预处理及统计可知，航班记录共 303 对，其中 49 对为宽体机，254 对为窄体机；登机口共 69 个，其中航站楼 T 有 28 个登机口，卫星厅 S 有 41 个登机口；旅客中转记录共 1703 条，旅客总数为 2833 人。计算结果表明：

（1）在 303 对航班记录中，有 254 对航班成功分配登机口，其中宽体机有 49 对航班分配成功，窄体机有 205 对航班分配成功，宽体机航班的分配成功率为 100%，窄体机航班的分配成功率 80.71%，总成功率 83.83%。

全部航班记录、窄体机航班、宽体机航班的成功分配数量如图 4-1 所示，其中每一柱由上至下的 2 个数字代表该类型总量、成功分配数量；全部航班记录、窄体机航班、宽体机航班的成功分配率如图 4-2 所示，其中每一柱由上至下的 2 个数字代表该类型总量、分配成功率。

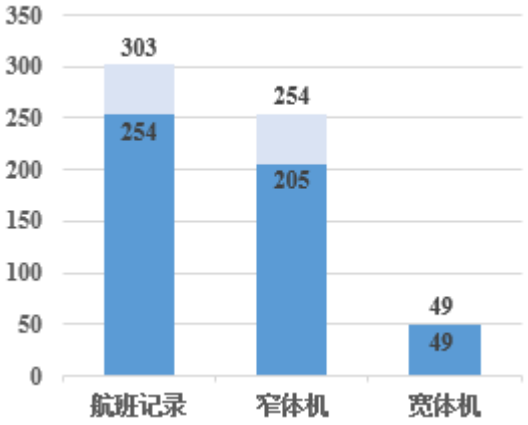

图 4-1　全部航班记录、窄体机航班、宽体机航班的成功数量

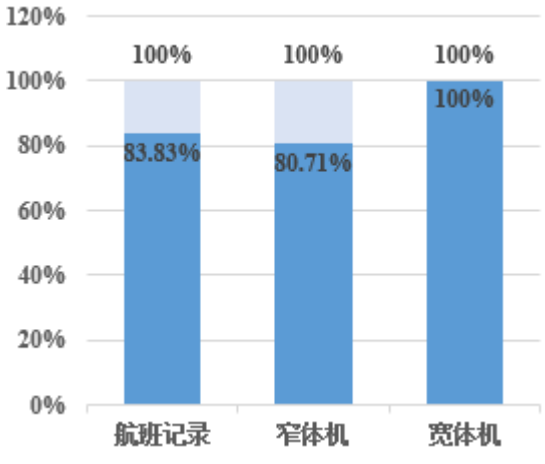

图 4-2　全部航班记录、窄体机航班、宽体机航班的成功率

（2）254 对成功分配的航班共停靠在了 67 个登机口，其中航站 T 楼使用了 28 个登机口，卫星厅 S 使用 39 个登机口；航站楼 T 使用的 28 个登机口共被使用 22340 分钟，平均使用率为 55.41%，卫星厅 S 使用的 39 个登机口共被使用 33185 分钟，平均使用率为 59.09%，故总使用的 67 登机口共被使用 55525 分钟，平均使用率为 57.55%。

问题 2 求解采用 C++编写主程序，程序文件详见附附件，程序模块介绍见附录 2。

总登机口、T 登机口和 S 登机口的使用数量如图 4-3 所示，其中每一柱由上至下的 2 个数字代表该类型总量、使用数量；总登机口、T 登机口和 S 登机口的平均使用率如图 4-4 所示，其中每一柱由上至下的 2 个数字代表该类型总量、平均使用率。

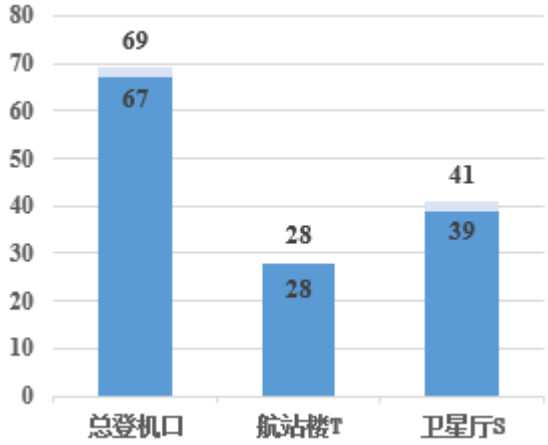

图 4-3　总登机口、T 登机口和 S 登机口的使用数量

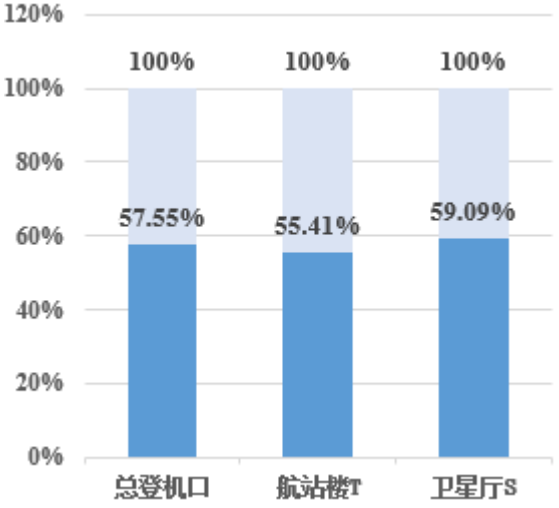

图 4-4　总登机口、T 登机口和 S 登机口的平均使用率

（3）旅客 2833 人中，2014 人的到达、出发航班都是成功分配登机口的航班，另外 819 人的航班分配至临时机位，换乘失败旅客为 **0**，失败率为 **0%**。（换乘失败：旅客换乘时间＞航班连接时间，且不考虑分配在临时机位的航班）。

5．问题 3 的模型建立与求解

5.1 问题 3 的分析

问题 3 是在问题 2 的基础上要求考虑中转旅客$T=\{t_1,t_2,\cdots,t_h\}$的换乘时间，且换乘时间需要考虑旅客乘坐捷运和步行时间。问题 3 的目标函数要在最大化分配到登机口的航班数量的基础上，最小化中转旅客的换乘紧张度，同时尽可能满足最大化未使用登机口数量。

同样，问题 3 也需要满足所有硬约束条件：唯一性约束；机体类别匹配约束；转场航班的到达类型、出发类型匹配约束；独占性约束；同一登机口安全时间间隔约束。

5.2 模型的准备与建立

5.2.1 数据定义

（1）变量

$tensity_u$——第u位旅客的换乘紧张度，$u=1,2,\cdots,h$

（2）其它数据说明

表 5-1 给出了中转旅客在不同场景下的最短流程时间和捷运乘坐次数，其中每一格的第一个数是最短流程时间（分钟），第二个数是捷运乘坐次数。

表 5-1 不同场景的最短流程时间和捷运乘坐次数

出发 / 到达		国内出发（D）		国际出发（I）	
		航站楼 T	卫星厅 S	航站楼 T	卫星厅 S
国内到达（D）	航站楼 T	15/0	20/1	35/0	40/1
	卫星厅 S	20/1	15/0	40/1	35/0
国际到达（I）	航站楼 T	35/0	40/1	20/0	30/1
	卫星厅 S	40/1	45/2	30/1	20/0

下面给出旅客换乘时间、航班连接时间、换乘紧张度的定义：

旅客换乘时间＝ 最短流程时间 ＋ 捷运时间 ＋ 行走时间，其中捷运时间为单程一次 8 分钟，行走时间如表 5-2 所示；

航班连接时间＝ 后一航班出发时间 － 前一航班到达时间；

换乘紧张度$tensity_u=\dfrac{\text{旅客换乘时间}}{\text{航班连接时间}}$，$u=1,2,\cdots,h$。其中计算$tensity_u$时，其对应的旅客换乘时间由该旅客的信息查表 5-1 所得，航班连接时间由该旅客信息中航班号对应的到达时间、出发时间作差所得。

表 5-2　行走时间表

	T-North	T-Center	T-South	S-North	S-Center	S-South	S-East
T-North	10	15	20	25	20	25	25
T-Center		10	15	20	15	20	20
T-South			10	25	20	25	25
S-North				10	15	20	20
S-Center					10	15	15
S-South						10	20
S-East							10

5.2.2　问题 3 的模型建立

问题 3 需满足的约束条件同问题 2：

（1）唯一性约束：$\sum_{j=1}^{m} x_{ij} \le 1$，$i=1,2,\cdots,n$

（2）机体类别匹配约束：$f_i^{Type}=g_j^{Type}$，$i=1,2,\cdots,n$，$j=1,2,\cdots,m$

（3）转场航班 i 与停靠登机口 j 的到达类型、出发类型匹配约束：$f_i^{Nation} \in g_j^{Nation}$，$i=1,2,\cdots,n$，$j=1,2,\cdots,m$

（4）独占性约束：$\sum_{k\in R_i} x_{ij}+x_{kj} \le 1$，$i,k=1,2,\cdots,n$，$j=1,2,\cdots,m$

问题 3 的目标函数要在最大化分配到登机口的航班数量的基础上，最小化中转旅客的换乘紧张度，同时尽可能满足最大化未使用登机口数量。旅客人数的数量级为数千级，由换乘紧张度的定义可知中转旅客总换乘紧张度的数量级为数千级，设计目标函数如下：

$$z_3=-\varphi\sum_{u=1}^{h} tensity_i+\omega\sum_{i=1}^{n}\sum_{j=1}^{m} x_{ij}+\rho\sum_{j=1}^{m} y_j$$

其中 φ，ω，ρ 为调节参数，为了平衡数量级本文取 $\varphi=1$，$\omega=100$，$\rho=0.1$。

综上，可以建立问题 3 的数学模型，见 $MOD3$。

$MOD3$

$$\max\ z_3=-\varphi\sum_{u=1}^{h} tensity_i+\omega\sum_{i=1}^{n}\sum_{j=1}^{m} x_{ij}+\rho\sum_{j=1}^{m} y_j$$

$$s.t.\begin{cases}\sum_{j=1}^{m} x_{ij}\le 1,\ i=1,2,\cdots,n\\ f_i^{Type}=g_j^{Type},\ i=1,2,\cdots,n,\ j=1,2,\cdots,m\\ f_i^{Nation}\in g_j^{Nation},\ i=1,2,\cdots,n,\ j=1,2,\cdots,m\\ \sum_{k\in R_i} x_{ij}+x_{kj}\le 1,\ i,k=1,2,\cdots,n,\ j=1,2,\cdots,m\end{cases}$$

5.3 模型的求解及分析

问题 3 的数学模型与问题 1 类似，问题 3 仅在目标函数增加了最小化中转旅客的换乘紧张度。问题 1 采用的遗传算法仍然使用于问题 3，故我们仍采用遗传算法。

5.3.1 算法设计

问题 3 同样采用遗传算法求解，算法的流程与问题 1 相似，分为个体编码设计、初始种群生成、赌轮选择、两点交叉、变异算子，其中问题 1 赌轮选择中的适应度函数不再适用问题 3，设计新的适应度函数以考虑中转旅客的最短流程时间，适应度函数为 $fitness = z_3$，其中 z_3 为问题 3 的目标函数值。

5.3.2 问题 3 求解结果及分析

对于问题 3 的求解，程序中对 1703 条旅客数据进行预处理：（1）由表 5-1 计算旅客换乘时间；（2）由到达航班所在的航班记录以及出发航班所在的航班记录计算航班连接时间。

问题 3 求解采用 C++编写主程序，程序文件详见附附件，程序模块介绍见附录 3。

按照 5.3.1 节所述的算法流程，我们进行了遗传算法的编程并求出了最优结果。由数据预处理及统计可知，航班记录共 303 对，其中 49 对为宽体机，254 对为窄体机；登机口共 69 个，其中航站楼 T 有 28 个登机口，卫星厅 S 有 41 个登机口；旅客中转记录共 1703 条，旅客总数为 2833 人。计算结果表明：

（1）在 303 对航班记录中，有 253 对航班成功分配登机口，其中宽体机有 49 对航班分配成功，窄体机有 204 对航班分配成功，宽体机航班的分配成功率为 100%，窄体机航班的分配成功率 80.31%，总成功率 83.50%。

全部航班记录、窄体机航班、宽体机航班的成功分配数量如图 5-1 所示，其中每一柱由上至下的 2 个数字代表该类型总量、成功分配数量；全部航班记录、窄体机航班、宽体机航班的成功分配率如图 5-2 所示，其中每一柱由上至下的 2 个数字代表该类型总量、分配成功率。

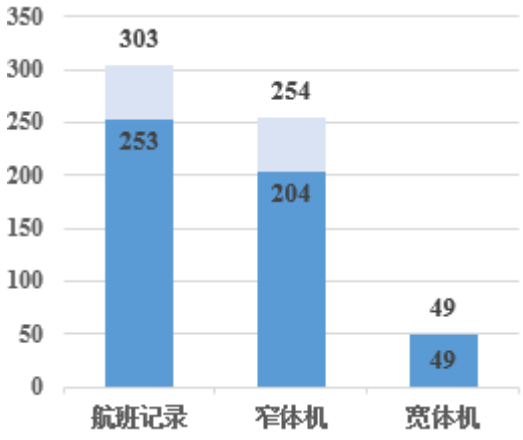

图 5-1 全部航班记录、窄体机航班、宽体机航班的成功数量

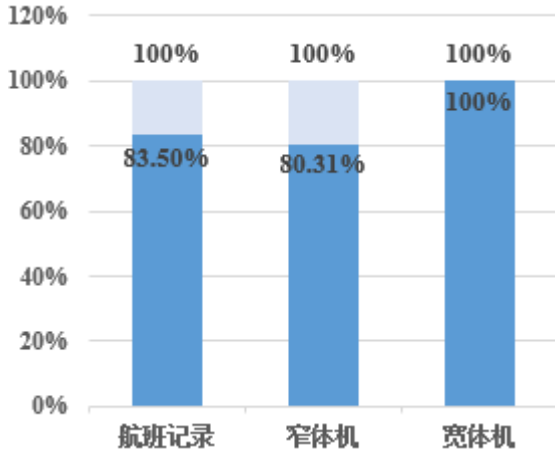

图 5-2 全部航班记录、窄体机航班、宽体机航班的成功率

（2）253 对成功分配的航班共停靠在了 66 个登机口，其中航站 T 楼使用了 27 个登机口，卫星厅 S 使用 39 个登机口；航站楼 T 使用的 27 个登机口共被使

用 22165 分钟，平均使用率为 57.01%，卫星厅 S 使用的 39 个登机口共被使用 33385 分钟，平均使用率为 59.45%，故总使用的 66 登机口共被使用 55550 分钟，平均使用率为 58.45%。

总登机口、T 登机口和 S 登机口的使用数量如图 5-3 所示，其中每一柱由上至下的 2 个数字代表该类型总量、使用数量；总登机口、T 登机口和 S 登机口的平均使用率如图 5-4 所示，其中每一柱由上至下的 2 个数字代表该类型总量、平均使用率。

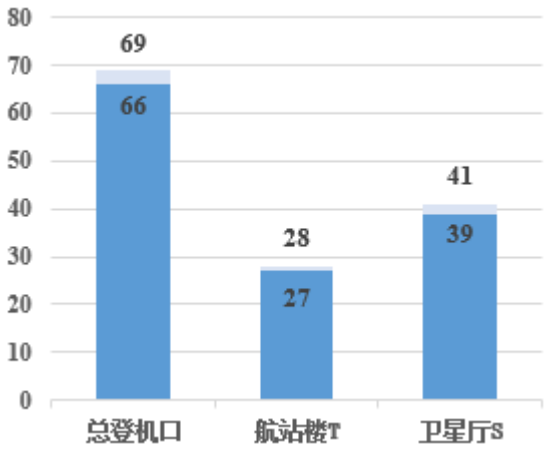

图 5-3　总登机口、T 登机口和 S 登机口的使用数量

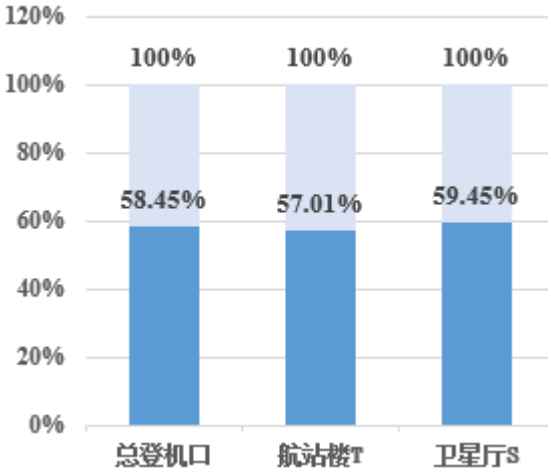

图 5-4　总登机口、T 登机口和 S 登机口的平均使用率

（3）旅客 2833 人中，1882 人的到达、出发航班都是成功分配登机口的航班，另外 951 人的航班分配至临时机位，换乘失败旅客为 0，失败率为 0%。（换乘失败：旅客换乘时间＞航班连接时间，且不考虑分配在临时机位的航班）。

（4）经计算，旅客最小换乘时间为 25 分钟，最大换乘时间为 81 分钟，以 5 分钟为间隔进行统计结果如表 5-3 所示。

表 5-3　旅客换乘时间统计表

旅客换乘时间（单位：分钟）	人数	累积人数	所占比率	累计比率
21-25	15	15	0.80%	0.80%
26-30	161	176	8.55%	9.35%
31-35	68	244	3.61%	12.96%
36-40	41	285	2.18%	15.14%
41-45	115	400	6.11%	21.25%
46-50	163	563	8.66%	29.91%
51-55	235	798	12.49%	42.40%
56-60	116	914	6.16%	48.57%
61-65	228	1142	12.11%	60.68%
66-70	237	1379	12.59%	73.27%
71-75	351	1730	18.65%	91.92%
76-80	44	1774	2.34%	94.26%
80-85	108	1882	5.74%	100.00%

旅客换乘时间分布图如图 5-5 所示。

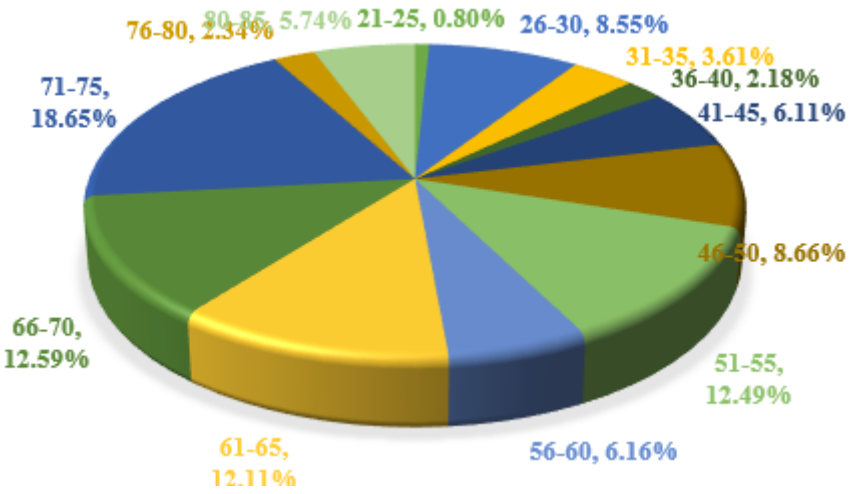

图 5-5　旅客换乘时间分布图

（5）经计算换乘紧张度，以 0.1 为间隔进行统计结果如表 5-4 所示。

表 5-4　换乘紧张度统计表

换乘紧张度	人数	累积人数	比率	累计比率
(0, 0.1]	0	0	0.00%	0.00%
(0.1, 0.2]	261	261	13.87%	13.87%
(0.2, 0.3]	519	780	27.58%	41.45%
(0.3, 0.4]	512	1292	27.21%	68.65%
(0.4, 0.5]	291	1583	15.46%	84.11%
(0.5, 0.6]	179	1762	9.51%	93.62%
(0.6, 0.7]	89	1851	4.73%	98.35%
(0.7, 0.8]	22	1873	1.17%	99.52%
(0.8, 0.9]	9	1882	0.48%	100.00%

旅客换乘紧张度分布图如图 5-6 所示。

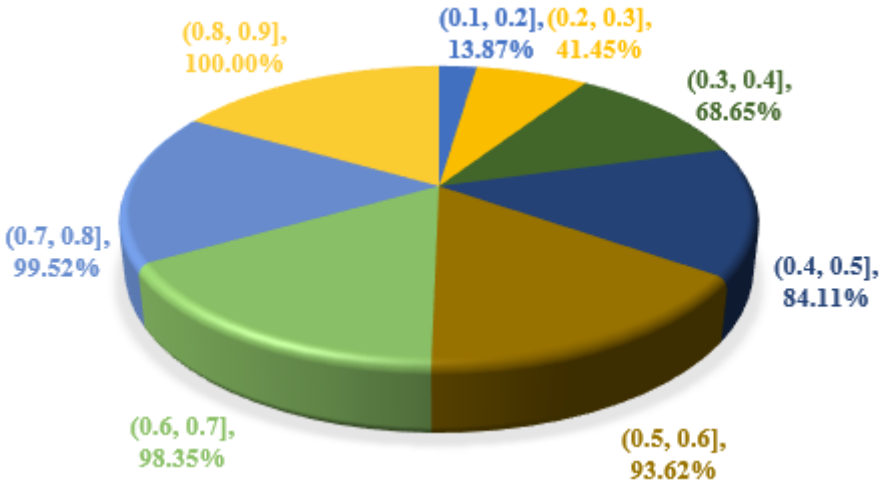

图 5-6　旅客换乘紧张度分布图

6．模型的评价

6.1 模型的优点

（1）采用 0-1 二维矩阵来表示航班－登机口的分配情况，简洁明了；

（2）定义了中间变量 R_i：与航班 i 登机口占用时间存在冲突的飞机集合：$\forall j \in \{1,2,\cdots,n\}$，若 $a_i < a_j \cap d_i + \alpha > a_j$ 或 $a_i > a_j \cap a_i < d_j + \alpha$，则 $a_j \in R_i$。R_i 同时结合了独占性约束、同一登机口安全时间间隔约束，简化了模型；

（3）模型的可移植性高，成功应用于遗传算法求解。

6.2 模型的缺点及改进方向

（1）问题 2 和问题 3 为了调节数量级的差距及目标间的优先，设计引入了数值参数 ω，ξ，ρ，φ，参数的取值要人为设定，这需要一定的先验知识，后续工作可以改善取值的方式如采取自适应等方法；

（2）本文所有模型对应的都是一个 NP 难问题，对精确方法求解做出了挑战，庆幸的是进化算法是当前一种解决此类问题的有效方法。

7．参考文献

[1] Shuo Liu, Wenhua Chen, Jiyin Liu, Optimizing airport gate assignment with operational safety constraints, 2014 20th International Conference on Automation and Computing, IEEE, 2014:61-66, 2014..

[2] Bouras A, Ghaleb M A, Suryahatmaja U S, The Airport Gate Assignment Problem: A Survey[J]. The Scientific World Journal, 2014(6):9165-9172, 2014.

[3] 王宇平，进化计算的理论和方法[M]，北京：科学出版社，21-28，2011。

[4] 李倩雯，机场停机位优化分配模型构建[D]，北京：北京交通大学，19-41，2018。

[5] 田晨，基于遗传算法的机场机位分配策略[J]，计算机工程，31(3)：186-188，2005。

8．附　　录

附录 1　问题 1 使用 C++编写主程序的模块介绍

（1）主函数：

```
int main(){
    ios_base::sync_with_stdio(0);
    cin.tie(0);
    srand(time(NULL));
    read();
    freopen("out1.txt","w",stdout);//结果输出到 out1.txt 文件
    CGA();
    return 0;
}
```

（2）结构体定义：

```
//种群个体用结构体表示
struct st{
    int dis[nf];
    double ans1,ans2;
    //dis[i]表示航班 i 分配的登机口号，ans1 标记未被占用的登机口，ans2 标记分配的航班对数量
};

//表示飞机的结构体，用来记录飞机属性
struct fly{
    string num,s;//num 表示航班记录，s 表示航班类型
    int di;//di 记录航班对<抵达类型，出发类型>数字化
    int dtime,atime;//抵达航班时间，出发航班时间
}fy[305];

//表示登机口的结构体，用来记录登机口属性
struct gate{
    string name,s;//name 表示登机口的名称，s 表示登机口的宽窄类型
    int di;//di 记录登机口<抵达类型，出发类型>数字化
}gt[75];
```

（3）程序各模块：

```
//比较函数
bool cmp(gate a, gate b)
bool cmp2(fly a,fly b)

//读入数据并进行转化处理
void read()
```

```
//判断飞机 i 和 j 是否冲突
void judgeff()

//判断飞机 i 和登机口 j 是否冲突
void judgefg()

//计算个体 a 的适应度
double fitness(st a)

//将初始化的排列序转化为种群个体
st exchange(st a)

//计算个体 a 的登机口未被使用的数量 a.ans1 和航班的分配数量 a.ans2
st fx(st a)

//初始化函数，生成初始化种群
st init()

//相当于变异算子，调整两点交叉后的个体，使其成为可行解
st adjust(st a)

//种群进化，进行选择交叉变异
st select_cross_mutation()

//遗传算法程序入口
void CGA()
```

附录 2　问题 2 使用 C++编写主程序的模块介绍

（1）主函数：

```
int main(){
    ios_base::sync_with_stdio(0);
    cin.tie(0);
    srand(time(NULL));
    read();
    freopen("out2.txt","w",stdout);//结果输出到 out2.txt 文件
        CGA();
    return 0;
}
```

（2）结构体定义：

```
//种群个体用结构体表示
struct st{
    int dis[nf];
    double ans1,ans2,fit;//fit 记录个体适应度
```

```
        //dis[i]表示航班 i 分配的登机口号，ans1 标记未被占用的登机口，ans2 标记分配的航班对数量
        int process_time;
        //process_time 用来记录最短流程时间
    };

    //表示飞机的结构体，用来记录飞机属性
    struct fly{
        string num,s;//num 表示航班记录，s 表示航班类型
        int di;//di 记录航班对<抵达类型，出发类型>数字化
        int dtime,atime,at2;//抵达航班时间，出发航班时间
    }fy[305];

    //表示登机口的结构体，用来记录登机口属性
    struct gate{
        string name,s;//name 表示登机口的名称，s 表示登机口的宽窄类型
        int di;//di 记录登机口<抵达类型，出发类型>数字化
    }gt[75];

    //表示旅客记录的机构体
    struct passenger{
        string name,num1,num2,t1,t2,fy1,fy2;//name 表示旅客记录，num1 表示抵达的航班号，num2 记录出发的航班号，t1 记录抵达航班日期，t2 记录出发航班日期，fy1 记录抵达飞机编号，fy2 记录出发飞机编号
        int people,afy,dfy,di;//people 表示旅客记录参与的旅客人数，afy 抵达飞机数字化，dfy 出发飞机数字化
    }pg[2005];
```

（3）程序各模块：

```
    //比较函数
    bool cmp(gate a, gate b)
    bool cmp2(fly a,fly b)

    //读入数据并进行转化处理
    void read()

    //判断飞机 i 和 j 是否冲突
    void judgeff()

    //判断飞机 i 和登机口 j 是否冲突
    void judgefg()

    //计算航班登机口在航站楼和卫星厅的转换
    int ts_number(st a,passenger b)
```

```
//计算个体适应度
double fitness(st a)

//计算最短流程时间
int p_time(st a)

//将初始化的排列序转化为种群个体
st exchange(st a)

//计算个体 a 的登机口未被使用的数量 a.ans1 和航班的分配数量 a.ans2
st fx(st a)

//初始化函数，生成初始化种群
st init()

//相当于变异算子，调整两点交叉后的个体，使其成为可行解
st adjust(st a)

//种群进化，进行选择交叉变异
st select_cross_mutation()

//遗传算法程序入口
void CGA()
```

附录 3　问题 3 使用 C++编写主程序的模块介绍

（1）主函数：

```
int main(){
    ios_base::sync_with_stdio(0);
    cin.tie(0);
    srand(time(NULL));
    read();
    freopen("out3.txt","w",stdout);//结果输出到 out3.txt 文件
    CGA();
    return 0;
}
```

（2）结构体定义：

```
//种群个体用结构体表示
struct st{
    int dis[nf];
    double ans1,ans2,fit;//fit 记录个体适应度
    //dis[i]表示航班 i 分配的登机口号，ans1 标记未被占用的登机口，ans2 标记分配的航班对数量
    int process_time;
    //process_time 用来记录最短流程时间
```

```
};

//表示飞机的结构体，用来记录飞机属性
struct fly{
    string num,s;//num 表示航班记录，s 表示航班类型
    int di;//di 记录航班对<抵达类型，出发类型>数字化
    int dtime,atime,at2;//抵达航班时间，出发航班时间
}fy[305];

//表示登机口的结构体，用来记录登机口属性
struct gate{
    string name,s,position;//name 表示登机口的名称，s 表示登机口的宽窄类型
    int di;//di 记录登机口<抵达类型，出发类型>数字化
}gt[75];

//表示旅客记录的机构体
struct passenger{
    string name,num1,num2,t1,t2,fy1,fy2;//name 表示旅客记录，num1 表示抵达的航班号，num2 记录出发的航班号，t1 记录抵达航班日期，t2 记录出发航班日期，fy1 记录抵达飞机编号，fy2 记录出发飞机编号
    double transfer_time;//转场时间
    int people,afy,dfy,di;//people 表示旅客记录参与的旅客人数，afy 抵达飞机数字化，dfy 出发飞机数字化
}pg[2005];
```

（3）程序各模块：

```
//比较函数
bool cmp(gate a, gate b)
bool cmp2(fly a,fly b)

//获得行走登机口类型
int getpos(gate a)

//读入数据并进行转化处理
void read()

//判断飞机 i 和 j 是否冲突
void judgeff()

//判断飞机 i 和登机口 j 是否冲突
void judgefg()

//计算航班登机口在航站楼和卫星厅的转换
int ts_number(st a,passenger b)
```

//计算个体适应度
double fitness(st a)

//计算最短流程时间
int p_time(st a)

//将初始化的排列序转化为种群个体
st exchange(st a)

//计算个体 a 的登机口未被使用的数量 a.ans1 和航班的分配数量 a.ans2
st fx(st a)

//初始化函数，生成初始化种群
st init()

//相当于变异算子，调整两点交叉后的个体，使其成为可行解
st adjust(st a)

//种群进化，进行选择交叉变异
st select_cross_mutation()

//计算转场时间
double transfer_tension(st a)

//遗传算法程序入口
void CGA()